CSC 3520
Machine Learning
Florida Southern College

HW4: Nearest Neighbors and Support Vector Machines
Due: Friday, December 9, 2022

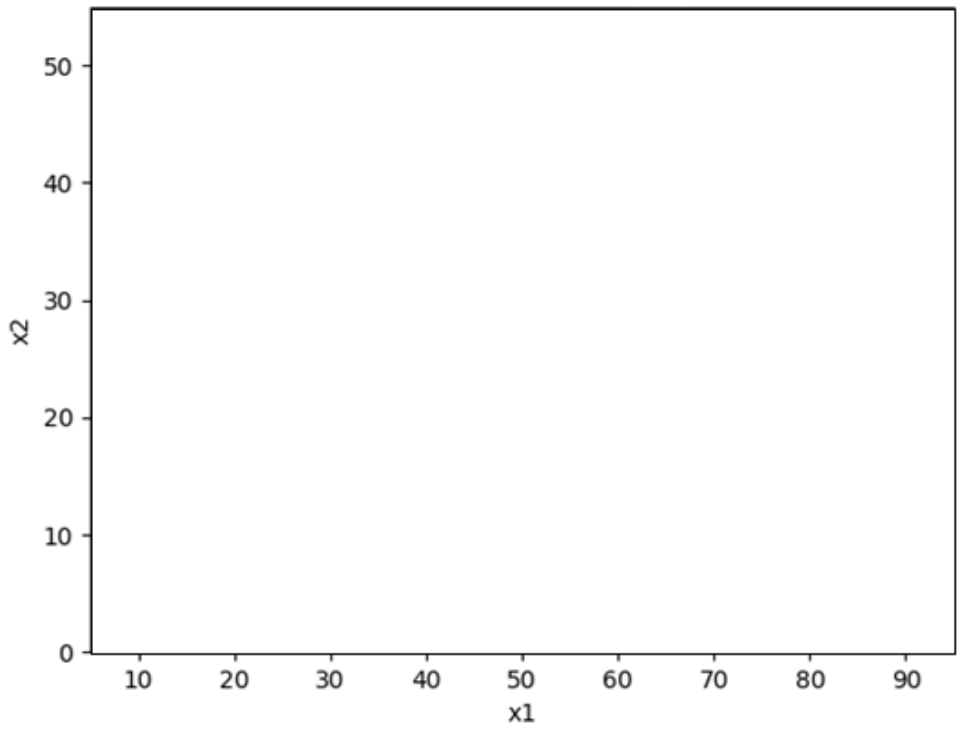
In this assignment, you will have the opportunity to:
(1) manually draw a Voronoi diagram and compute nearest neighbors for sample data
(2) write Python code to apply k-nearest neighbors to a large, real-world dataset
(3) write Python code to apply support vector machines to a large, real-world dataset
(4) practice comparing machine learning algorithms

1. **Nearest Neighbors.** Suppose you are given the follow 2D dataset:

$$X = \begin{bmatrix} 90 & 35 \\ 70 & 5 \\ 35 & 50 \\ 10 & 50 \\ 50 & 30 \end{bmatrix} \qquad Y = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

where X are the features (x_1, x_2), Y are the labels, each row is a unique training sample, and there are two classes [0, 1].

(a) Plot the data (by hand) below. Use 'x' for $Y = 0$ and 'o' for $Y = 1$.



(b) On the same plot, draw the complete Voronoi diagram; that is, draw the appropriate lines based on Euclidean distance that partition the 2D space into regions such that each data point has its own region.

(c) Using a **bold** line, identify the decision boundary on the Voronoi diagram.

(d) Now suppose you are given three test samples:

$$X_{test} = \begin{bmatrix} 15 & 5 \\ 30 & 40 \\ 80 & 20 \end{bmatrix}$$

Add each test sample to the plot using '▲'.

(e) Compute the **Manhattan distance** between each test and training sample. Fill out the table below.

		Training Data				
		1	2	3	4	5
Test Data	1					
	2					
	3					

(f) Using the distance information in the table above, determine the class label (0 or 1) for each test sample using *k*-nearest neighbors. Fill out the table below, where **each column is a different value for *k***. In the case of a tie, choose the class with the higher prior probability.

		k				
		1	2	3	4	5
Test Data	1					
	2					
	3					

2. **Handwritten Digit Recognition.** The goal of this problem is to train multiple machine learning classifiers on the same real-world dataset for comparison.

   (a) Write a Python program (`kdigits.py`) that applies *k*-nearest neighbors to the MNIST handwritten digit dataset. Use $k = 3$. Train the classifier on all 60,000 training samples. Test the classifier on the first 100 testing samples. Print the test accuracy.

   Find a misclassified test sample. Show the test image alongside the 3 nearest neighbors from the training set. Does it make sense why the classifier predicted the wrong digit?

   (b) Write a Python program (`sdigits.py`) that applies support vector machines to the MNIST handwritten digit dataset. Feel free to vary the SVM parameters (e.g. kernel, soft margin). How few training samples do you need to get reasonable performance?

   HINTS:
   - Try loading the dataset directly from `Keras` using the following code:
     ```
     from tensorflow.keras.datasets import mnist
     (xtrain, ytrain), (xtest, ytest) = mnist.load_data()
     ```
   - Try using `np.where` to find incorrectly-labeled samples
   - Try using `matplotlib.pyplot` and `scikit-image` to show images (`skimage.util.montage`)